MACHINE LOADING IN FLEXIBLE MANUFACTURING SYSTEM USING ARTIFICIAL IMMUNE SYSTEM

A THESIS SUBMITTED IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE IN BACHELOR OF TECHNOLOGY

IN

MECHANICAL ENGINEERING

BY

TANMAYA MISHRA

ROLL NO. 107ME012

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA.

2011.

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA

CERTIFICATE

This is to certify that the thesis entitled, ―Machine Loading in Flexible Manufacturing system using Artificial Immune algorithm by Tanmaya Mishra is in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Mechanical Engineering at National Institute of Technology, Rourkela (Deemed University), is an authentic work carried out by him under my supervision. To the best of my knowledge the matter embodied in the thesis has not been submitted to any University/Institute for the award of any Degree or Diploma.

Date: 11 May, 2011

Prof. S.S. Mahapatra

Department of Mechanical Engineering

National Institute of Technology Rourkela

ACKNOWLEDGEMENT

I avail this opportunity to extent my hearty indebtedness to my guide ―Prof. S.S. Mahapatra‖ , Mechanical Engineering Department, for his valuable guidance, constant encouragement and kind help at different stages for the execution of this dissertation work. I also express my sincere gratitude to ―Prof. R.K. Sahoo‖ , Head of the Department, Mechanical Engineering, for providing valuable departmental facilities and ―Prof. S.K Sahoo, Prof. C.K Biswas and Prof K.P Maity for constantly evaluating me and for providing useful suggestions.

Submitted By:

Tanmaya Mishra

Roll No. 107ME012

Mechanical Engineering

National Institute of Technology Rourkela

Rourkela-769008

Contents

Abstract

In this project thesis the FMS loading problem is discussed with the objective to minimize the system unbalance and throughput by the use of Artificial Immune system. Manufacturing technology focuses primarily on flexibility and productivity. With the product variety and product life being the characterizing standards it is important that the flexibility of the job shop is maintained as its efficiency is increases. The complexity of a basic Machine loading problem in FMS is very high due to the different flexibility criteria as Part selection, Operation allocation and the various constraints involved. This dissertation proposes a soft computing technique with constraints on tool capacity and workload of the machine. The aim of using this algorithm is to reach an optimal solution and to ease the tedious computations in large problems involving loading which are NP hard problems. Immune algorithm is a very suitable method due to its self learning and memory acquisition abilities. First some sample machine loading problems are collected from the literature and the optimal system unbalance of the machine is calculated using LINGO optimization software. This project improves some issues inherent in existing techniques and proposes an effective Immune algorithm with reduced memory requirements and reduced computational complexity. The proposed Algorithm is tested on 3 problems adopted from literatures and the results reveal substantial improvement in solution quality over the existing basic mathematical approaches.

INTRODUCTION

Productivity and Flexibility, these are the primary goal of today's production technology. They can only be achieved in fully integrated manufacturing environments. A flexible manufacturing system (FMS) is an integrated computer-controlled configuration which consists of numerical control (NC) machine tools, auxiliary production equipment and a material handling system (MHS).It is designed to simultaneously manufacture a low to medium volumes of a wide variety of high quality products at low cost.

The numerically controlled units include NC machines of different kinds, AGV vehicles, coordinate measuring equipments and robots. FMS ensures quality product and maintains a small lead time hence reducing the time to market. The objectives of FMS mainly includes in achieving efficiency in a balanced automated high volume mass production and in low volume job-shop production (Groover and Zimmer, 1986). Use of fixtures and tool magazines practically eliminates setup time. Mukhopadhyay and Tiwari (1995) have developed a method of grouping part operations and tools on the principle of conjoint measurement and solved the machine loading problem by minimizing the maximum difference between machine utilization called system over utilization time. These features permit economic production of a large variety of parts in low volumes.

Loading decisions play crucial role in inducing such behavior by processing the job in a feasible sequencing schedule. Effective loading decisions are particularly important in the large and complex manufacturing systems. . According to Stecke (1983), machine loading problem is one of six post release decisions of a flexible manufacturing system that is known for its computational complexity and high variability. FMS planning consists of pre-release and post-

release decisions. The pre-release decision problem includes prearrangement of parts and tools before the main processes of an FMS. FMS scheduling or post-release decision considers sequencing and routing of part types, when the system is in progress. Stecke(1983), Sarin and Chen (1987) categorizes the pre release decision problems into six types: (a) machine grouping, (b) part-type selection, (c) batching of part type, (d) production rate determination, (e) resource allocation, and (f) loading. Hwang (1986) investigates the production-planning problem and finds that the two sub-problems—part selection and machine loading—are crucially important.

Grouping of resources, selection of part mixes, aggregate planning are the decision levels that provide input to the machine loading decision which henceforth provide inputs to succeeding processes of scheduling and control. In this problem machine loading problems are considered in FMS environment. The unique characteristic that distinguishes FMS from other factory automation technologies is the ability to achieve flexible automation i.e., the capacity to efficiently produce a great variety of part types in variable quantities.

LITERATURE REVIEW

1. Objectives in Loading.

a) Stecke(1983) studied and described six main objectives in machine loading.

1. Balancing the machine processing time.
2. Minimizing the number of movements.
3. Balancing the workload per machine for a system of groups of pooled machines of equal sizes.
4. Unbalancing the workload per machine for a system of groups of pooled machines of unequal sizes.
5. Filling the tool magazines as densely as possible.
6. Maximizing the sum of operations priorities.

b) (Stecke, 1983; Shanker and Srinivasulu, 1989).Ammons et al. (1985) resolves the loading problem considering a bi-criteria objective of balancing workload and minimizing work stations visits.

c) Tiwari et al. (1997) and Mukhopadhyay et al.(1992) tackle machine-loading problem using heuristic approaches with an objective of minimizing system unbalance and maximizing throughput.

Objective functions used in this thesis are the minimization of system unbalance and the maximization of throughput for the following reasons:

(1) Minimization of system idle time leads to higher machine utilization,

(2) One of the most important goals of any loading policy is enhancing total system output, which is the same as throughput,

(3) Kim and Yano (1997) have found that throughput maximization by balancing the workloads on the machine often results in limiting the tardiness

2. Solving Approaches to the Loading Problem.

Machine-loading problem can be addressed mainly by four approaches:

a) ***Heuristic oriented methods.***

b) ***Optimization-based methods or Mathematical programming approaches.***

c) ***Multi-criteria decision-making approaches.***

d) Simulation based approaches.

a) <u>Mathematical programming approaches</u>

i) The first mathematical formulation for grouping in FMS loading was given by Stecke as non linear 0-1 mixed integer programmes. It was assumed that product mix problem is already solved and so the model is left suitable limited to only dedicated FMS.

ii) O'Grady and Menon employed a goal programming model for loading a real-life FMS.

iii) Berrada and Stecke developed a branch-and-bound algorithm for balancing workloads on machines.

iv) Guerrero et al. developed mixed-integer linear program considering alternative routes for each part type. It directly determined the optimal number of copies of each tool type to be loaded into each tool magazine.

b) Multi-criteria decision-making approaches.

i) Ammons et al. developed a bicriterion objective for the loading problem, i.e., balancing workloads and minimizing visits to the workstations.

ii) Shanker and Tzen addressed the machine-loading problem in random FMS with the bi-criterion objective of meeting the due dates of the jobs and balancing the workload amongst the machining centers. They formulated a simulation model and examined the effects of loading on system performance under different dispatching rules.

iii) Swamkar and Tiwari addressed machine-loading problem of an FMS having the bicriterion objectives of minimizing system unbalance and maximizing the throughput. A hybrid algorithm based on tabu search and simulated annealing (SA) was employed to solve the problem. The main advantage of this approach is that it uses a short-term memory provided by the tabu list to avoid revisiting the solution while preserving the stochastic nature of the SA method

c) Simulation based approaches.

i) Stecke and Solberg had carried out a simulation study for dedicated type FMS examining five loading strategies versus 16 dispatching

rules. As their study was based on a dedicated type of FMS, the loading strategies were simply the procedures to allocate operations to a number of similar machines.

ii) Gupta et al. described a dispatching approach for FMSs where all parts were stored in a central buffer. Parts were selected according to pre-determined loading rules. Simulation experiments showed that proposed dispatching approach outperformed the traditional one with respect to make-spans, average flow time and average tardiness.

d) Heuristic oriented methods.

i) Mukhopadhyay et al. developed a heuristic solution to the loading problem in FMS by developing the concept of essentiality ratio for the objective of minimization of system unbalance and maximizing the throughput.

ii) Tiwari et al. used fixed pre-determined job ordering/job sequencing rule as input to their proposed heuristics of perturbation scheme known as 'modified insertion scheme' for generating new job sequences.

iii) Vidyarthi and Tiwari proposed a fuzzy-based methodology to solve machine-loading problem in FMS. Honghong Yang and Zhiming Wu developed a mixed integer-programming model that integrates part type selection and machine loading together.

3. AIS and Evolutionary Computing.

Evolutionary Computing (EC) was initially an approach to Artificial Intelligence (AI) which has been further used to solve numerical and combinatorial optimization problem. It includes Evolutionary programming, genetic programming, Genetic algorithm ((Fozel and Corne, 2003), Immune algorithm. However all these techniques are related.

i) DeCastro and Zuben (2002) have reviewed the Clonal selection concept together with hypermutation operator to develop a computation tool named CLONALG (here denoted by CA).

ii) Cutello et al. (2002) have proposed opt-IA, a modified version of CA, by using three immune operators i.e. cloning, hypermutation and aging operator.

iii) Cutello et al. (2006) , Eiben and Schoenauer (2002), Muller et al. (2002) introduced a new and improved version of opt-IA, having features viz. real coding representation, cloning operator, inversely proportional hypermutation and aging operator.

iv) Kumar and Shanker solved part type selection and machine-loading problems in production planning of FMS by using genetic algorithm (GA).

LOADING IN FMS AND AIS.

1 . Flexible manufacturing system

A flexible manufacturing system is a highly automated GT machine cell consisting of a group of processing work stations (CNC M/C tools), interconnected by an automated material handling and storage system and controlled and distributed by a computer system. It uses principles of group technology. The various types of FMS are

- A single machine cell
- A flexible manufacturing cell
- A flexible manufacturing system

Flexibility of an FMS depends on

- Ability to identify and distinguish among different product styles or incoming parts to be processed.
- Quick changeover of operating instructions.
- Quick change over of physical set-up.

The categories for flexibility are

- ***A dedicated FMS*** which produces a limited variety of work and completes universe of parts to be made is known as advance FMS. The part family is likely to be based on product commonality rather than geometric similarity. Product design is considered stable.

- ***A random order FMS*** is more appropriate when part family is large, there are substantial variation in part configuration, new part design is introduced to the system with changes in parts currently produced with change in production schedule

Flexibility criteria

- Part variety
- Schedule change
- Error recovery
- New part

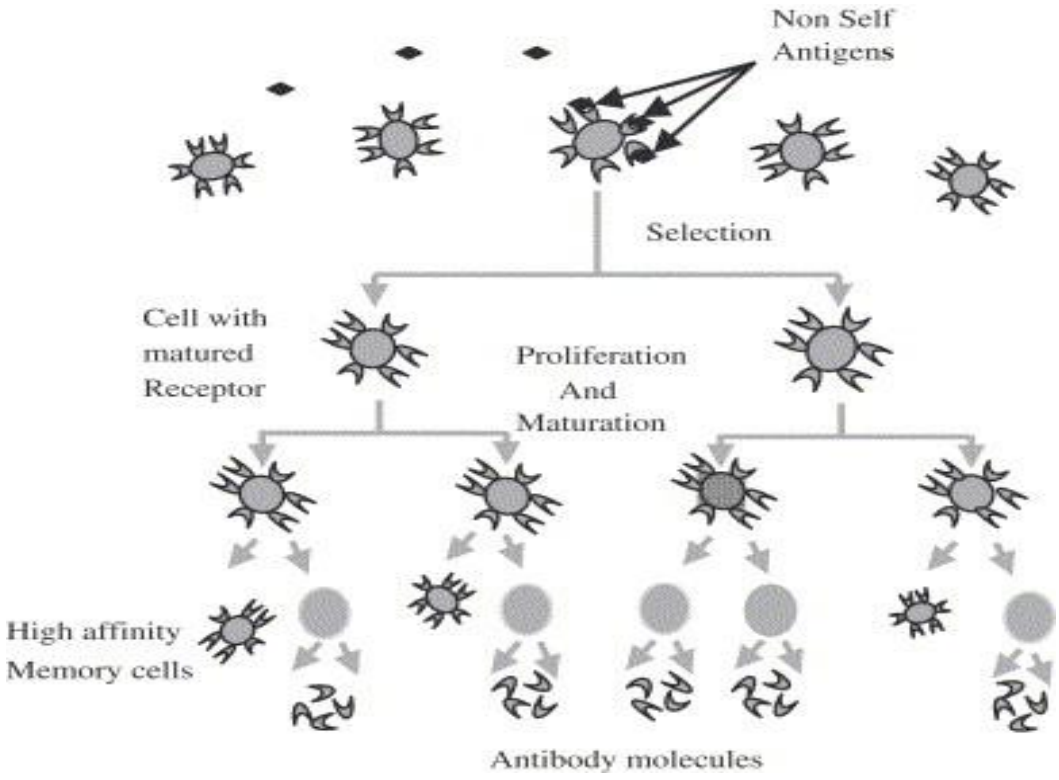
  For many industries it is accepted that reduction in time to market of one week can provide significant revenues. Currently firms apply Flexible manufacturing systems to reduce their Time to market. One of the main purposes of FMS is to achieve efficiency of a well balance transfer line while retaining flexibility of a job shop.

2. Artificial Immune system and Chaotic generators.

a) *Biological Inspiration*

The function of biological IS is to protect the body from the foreign matters, more known as antigens. Antigens stimulate the antibodies that reside in the body. The key roles of antibodies are to identify, bind and eliminate the antigens. Clonal selection explains the response of IS, when a non-self antigen pattern is recognized by the B-cells. It is selected to proliferate and produce antibodies in high volume by cloning. The new clonal cells undergo hypermutation for improving antibodies affinity that leads to antigenic neutralization and elimination (Dasgupta, 1999). The overall procedure of clonal selection is schematically shown in Fig. 1

Fig. 1. Process of clonal selection, proliferation and affinity maturation



Random number generators (RNGs) have been widely used in algorithms for generating randomness, and every kind of probabilistic distribution. However, RNGs are notoriously known for having slow convergence and an inherent characteristic of sticking to a solution. To overcome this difficulty chaotic generators have recently been used to generate the random numbers instead of RNG

b) Random Number Generators

Random Number generators

The computer generated random numbers are called pseudo random number. They are algorithm that can create long runs of number with good random properties but eventually the sequence repeats.

The string values generated by such algorithm are generally determined by a fixed number called seed.

$$x_{N+1} = (ax_N + b) \bmod m$$

The above is the Linear congruential generator. The maximum number of numbers the formula can generate is the modulus, m.

The seed value X_N belongs from zero to 1.

If the frequency of seed value is taken on a graph it basically is productive and hence the direction of convergence is known and hence convergence is slow. There is normal probabilistic distribution.

Chaotic generators

Chaotic sequences are the type of random number generator (RNG) whose choice is justified by their ergodic and stochastic behavior (spread spectrum characteristic) (Determan and Foster, 1999). Due to unpredictability in chaotic sequence, it has been applied in various fields, such as secure transmission, natural modeling phenomena, etc.

Caponetto et al. (2003) utilized the chaotic sequences in different phases of evolutionary algorithm such as creation of individual present in a population set, selection of potential individuals from a population set, introducing the random changes into the individual present in

the population, etc. The reason behind opting the chaotic sequences is due to their ability to converge fast toward optimal solution, while retaining a proper balance between exploitation and exploration (Chen and Aihara, 1995; Luo and Shao, 2003).

The logistic map (Caponetto et al., 2003)

$$N(T+1) = R*N(T)*[1-N(T)]$$

Where, *N(t) is the value of chaotic variable in t th iteration and*

R shows the bifurcation parameter of the system.

The logistic operator is iterated with initial value N(0)=0.1 and R=4 for 400 iterations.

From this figure, it is evident that the spread spectrum characteristic of logistic mapping enables it to be utilized in place of RNGs.

3. Proposed Immune Algorithm

a) ARTIFICIAL IMMUNE SYSTEM.

The artificial Immune system is built around the two principles of immune system.

i) Clonal selection principle

ii) Affinity maturation principle

i) Cloning selection principle

Each sequence(antibody) has a makespan(overall completion time) value which refers to the affinity of the antibody. Affinity value of each sequence is calculated from affinity function given as :

$$affinity(p) = 1/makespan$$

So a lower makespan value gives higher affinity value. Further cloning in antibodies is done directly proportional to their affinity function values. So antibodies with lower makespan values will generate more clones. An affinity function is defined based on the makespan value of the sequence.

ii) Affinity Maturation Principle

It consists of two methods namely mutation and receptor editing.

1) **Mutation :** A two phased mutation procedure were used for the generated clones.

(a) **Inverse mutation**: For a sequence s, let i and j be randomly selected two positions in the sequences. A neighbor of s is obtained by inversing the sequence of jobs between i and j positions. If the makespan value of the mutated sequence (after inverse mutation) is smaller than that of the original sequence (a generated clone from an antibody), then the mutated one is stored in the place of the original one. Otherwise, the sequence will be mutated again with random pair wise interchange mutation.

(b) **Pair wise interchange mutation**: Given a sequence s, let i and j be randomly selected two positions in the sequences. A neighbor of s is obtained by interchanging the jobs in positions I and j. If the makespan value of the mutated sequence (after pair wise interchange mutation) is smaller than that of the original sequence, then store the mutated one in the place of the original one. In the case where the algorithm could not find a better sequence after the two-mutation procedure, then it stores the original sequence (generated clone).

2) ***Receptor editing:*** After cloning and mutation processes, a percentage of the antibodies (worst %B of the whole population) in the antibody population are eliminated and randomly created antibodies are replaced with them. This mechanism allows finding new schedules that correspond to new search regions in the total search space.

3) PROPOSED ARTIFICIAL ALGORITHM

All of these algorithms require a set of population for initialization; they need to evaluate the individual fitness value to suit the selection process. Generic materials are exchanged in the recombination operators to emphasize the variation in the solution string. Finally the process is iterated to get the optimal solution.

Procedure

Initialize population (randomly)

Individuals (candidate solution)

Evaluation (fitness function) for all antibodies

While (termination criterion not satisfied)

Select (superior antibodies from parent population)

Cloning based on fitness value

Variation operators on clones (*Hypermutation*)

Evaluate new generated antibodies

Selection of superior antibodies

Creation of next generation population

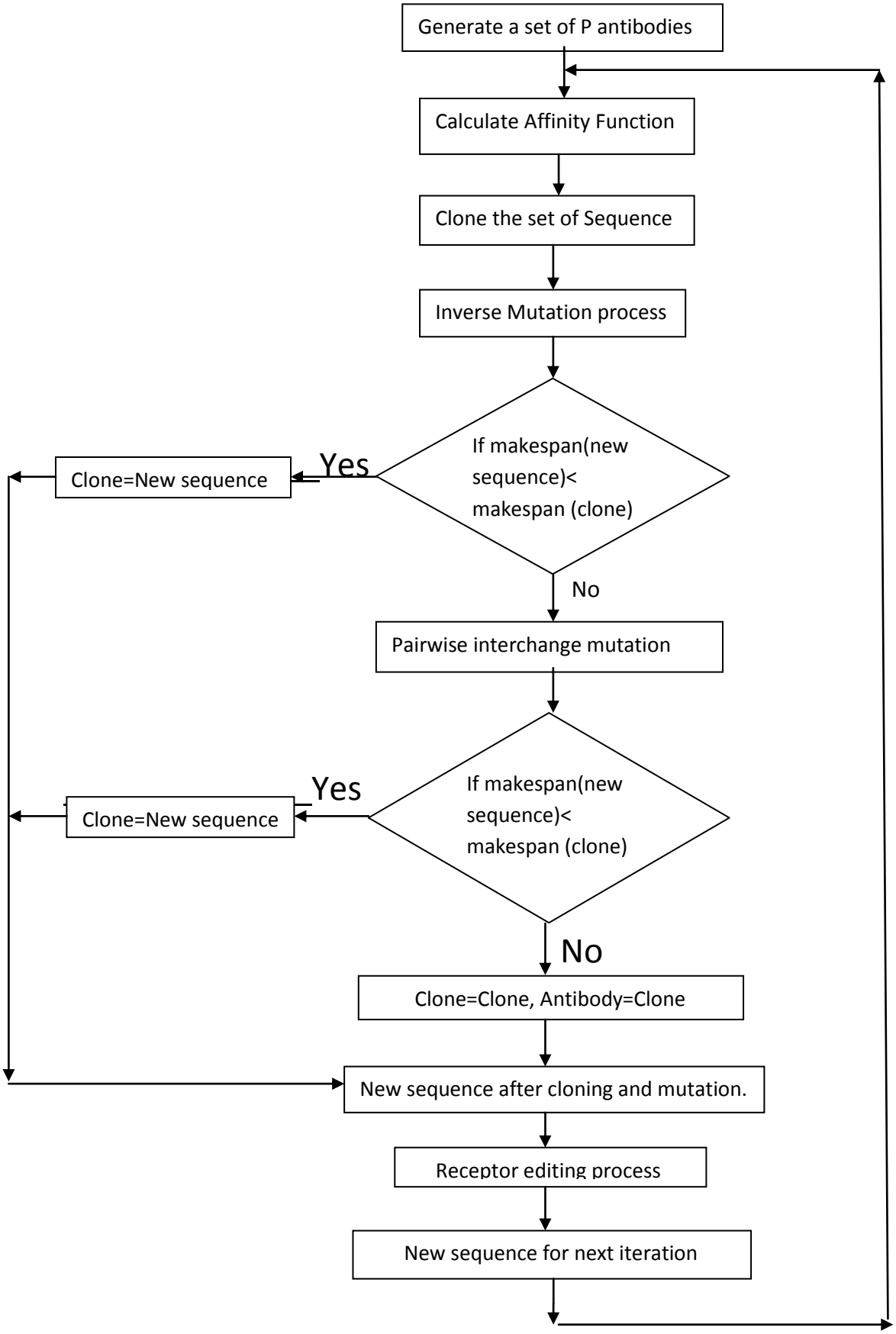
End

Chaotic sequences are the type of random number generators that is characterized by its ergodic and stochastic behavior (spread spectrum characteristic). Such sequences have been adopted in evolutionary algorithms for random number generation utilized in different phases of algorithmic computation such as:

• Initialization of a population.

• Selection of potential individuals from the set of population.

• Introducing random changes into the individuals present in the population.



Flowchart For Artificial Immune Algorithm for Machine Loading[Ref 10]

Problem description

To analyze a machine-loading problem for a random FMS, this work considers multiple machines and a fixed number of tool slots. In a given planning horizon, part types arrive randomly and their operation times and tool slot requirements are known.

The random FMS, considered here, is capable of performing operations that may be either essential or optional. Essential operations are those operations that can be done only on specific machines using specific tool slots whereas optional operations are flexible and can be carried out on one or more machines interchangeably. Therefore, it makes intuitive sense to preserve the optional operations as long as possible while considering all possible routes. It is instructive to note that flexibility lies in the selection of a machine for processing the optional operations for a particular part. The FMS under consideration is able to elicit the flexibility pertaining to selection of a machine, processing of an operation, selection of part-type sequence, etc.In a given planning horizon, machine-loading problem deals with choosing a part type from a pool of part types, and allocates its operation to appropriate machines achieving desired system performance measures. This is done taking into account the technological and capacity constraints. Let us consider an example FMS in which six part types are to be processed on four machines, each having three tool slots and different processing times for every operation. Each part type consists of two operations, which can be performed, on any of the machines without altering the sequence of operations. The adaptability of each machine to perform many different operations allows several operation assignment possibilities generating alternative part routes.

Thus, there can be a fairly large number of combinations in which operations of the part type can be assigned on the different machines while satisfying all the technological and capacity constraints. Further consideration of flexibilities such as: tooling flexibilities, part movement

flexibilities, etc., along with the constraints of the system configuration and operational feasibility make the problem even more complex.

These operation–machines allocation combinations are evaluated using two common performance measures: system unbalance and throughput. System unbalance is the sum of unutilized or over-utilized time on all machines available in the system. Maximization of machine utilization is identical to minimization of system unbalance, whereas _throughput_ refers to the units of part types produced. In the worst case, to arrive at an optimal or a near-optimal solution for the machine-loading problem, it may be necessary to explore each combinatorial allocation with respect to a given objective function (minimization of system unbalance or maximizing throughput), and simultaneously satisfying all the constraints.

The above problem is addressed considering the following objective functions:

(1) minimization of system unbalance;

(2) maximization of throughput;

(3) multiple objectives: a combination of minimization of system unbalance and maximization of throughput.

1. Modeling the loading problem

1.1 Notations and characteristics

m: number of jobs available for loading

n: number of machines available in the system

T_j*: length of scheduling period for jth machine*

p_{ikj}*: processing time of operation k on job i in machine*

y_i*: number of operations on job i*

a_i*: batch size of job i*

B(i,k): set of machines on which operation k of job i can be performed

S_{ikj}*: number of tool slots required for processing operation k of job I on machine j*

t_j*: tool slot capacity of machine j*

1.2 Decision variables

m = 6 , n = 4 , y_i = 2 for all i

So, number of variables = $m \times n \times y_i + x_i$

Number of constraints = $n + m \times y_i + m$

$\mathbf{x_i}$ = { 1 , if job i is selected

{0 , otherwise

$\mathbf{x_{ikj}}$ = { 1 , if operation k of job i is assigned to machine j

{0 , otherwise

7.3 Objective function

Minimize , f , system unbalance

$$f = \sum_{j=1}^{n} \left| T_j - \sum_{i=1}^{m} \sum_{k=1}^{y_i} a_i p_{ikj} x_{ikj} \right| \quad \longrightarrow \quad 1(a)$$

Subject to the following constraints

a) (A technological constraint)Only available tool slots can be used

$$\sum_{i=1}^{m} \sum_{k=1}^{y_i} s_{ikj}\, x_{ijk} \le t_j \qquad \text{for all, } j=1,2,\ldots\ldots,n$$

b) A particular operation of a job is done only in one machine

$$\sum_{G \in B(i,k)} x_{ikG} \le 1 \qquad \text{for all } i=1,2,\ldots\ldots,m \text{ and for all } k=1,2\ldots\ldots,y_i$$

c) A job cannot be split

$$\sum_{k=1}^{y_i} \sum_{j=1}^{n} x_{ikj} = x_i y_i \qquad \text{for all } i=1,2,\ldots\ldots m$$

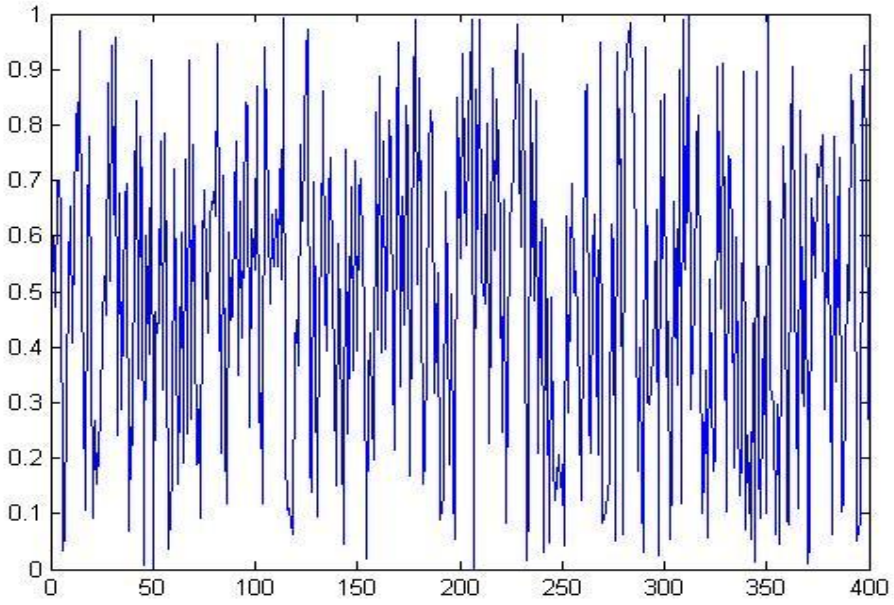
RESULTS AND DISCUSSIONS

- 1. A comparative account between use of random number generator and chaotic number generetors to be used in receptor editing. When logistic mapping is done of the chaotic number there is a large variation and unpredictive sequence generation, but a faster convergence towards the solution.
- 2. Objective functions was written for various problem statements and the minimum system unbalance was calculated for the machines of the FMS using LP programming software LINGO.
- 3. A C code was written following the artificial Immune Algorithm and the sample problems of the LINGO were used to calculate the output of System Unbalance and Throwput.

Conclusion

The main contribution of this paper is to develop an efficient evolutionary algorithm based on multi-stage programming approach to solve machine-loading problem of random FMS. The proposed IA enhances the applicability of traditional clonal algorithm by making some modifications in the operators. Maximum possible throughput and minimum possible system unbalance has been achieved. The comparisons show the supremacy and robustness of the proposed algorithm over normal mathematical optimization methods.

Numbers generated randomly by Random Number generators from 1 to 400



Numbers generated by chaotic number generators from 1 to 400

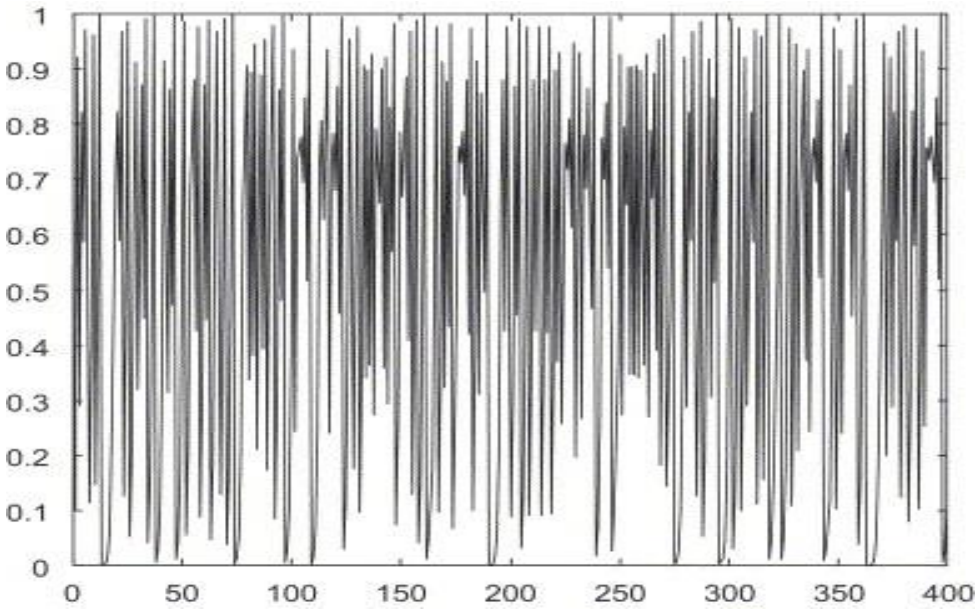


Table 1. Problem 1

Jobs(i)	Batch size(b)	Operation Number(k)	Machine Number(j)	Total processing time(in mins)(t)	Tool slots available($t_{j)}$
1	15	1	4 2	10 12	2 2
2	10	1	1	20	1
		2	3	35	2
3	12	1	1	22	3
4	9	1	3 2	25 25	1 1
5	16	1	4 2 3	30 25 27	2 1 2
		2	1 4	16 16	1 1
6	11	1	2	21	3

Solution In lingo.

Linearization components added:
Constraints: 16
Variables: 16
Integers: 4

Global optimal solution found.
Objective value: 171.0000
Objective bound: 171.0000
Infeasibilities: 0.000000
Extended solver steps: 0
Total solver iterations: 8

Model Class: MILP

Total variables: 27
Nonlinear variables: 0
Integer variables: 4

Total constraints: 33
Nonlinear constraints: 0

Total nonzeros: 84
Nonlinear nonzeros: 0

Variable	Value	Reduced Cost
X211	1.000000	0.000000
X311	1.000000	0.000000
X521	0.6250000E-01	0.000000
X112	0.000000	175.0000
X412	0.4222222	0.000000
X512	0.8125000	0.000000
X612	1.000000	0.000000
X223	1.000000	0.000000
X413	0.5777778	0.000000
X513	0.000000	32.00000
X114	1.000000	0.000000
X514	0.1875000	0.000000
X524	0.9375000	0.000000

Row	Slack or Surplus	Dual Price
1	171.0000	-1.000000
2	0.9375000	0.000000
3	0.7652778	0.000000
4	2.422222	0.000000
5	1.687500	0.000000
6	0.000000	0.000000
7	0.000000	183.3333
8	0.000000	0.000000
9	0.000000	0.000000
10	0.000000	0.000000
11	0.000000	0.000000
12	0.000000	186.6667
13	0.000000	0.000000
14	0.000000	-125.0000
15	0.000000	-350.0000
16	0.000000	-264.0000
17	0.000000	-225.0000
18	0.000000	-400.0000

Sample Problem 2

Job(i)	Batch size(b)	Operation Number(k)	Machine Number(j)	Unit Processing Time(t)	Available tool slots(t_j)
1	14	1	4	10	1
		2	3 1 2	26 26 26	2 2 2
		3	3	16	1
2	12	1	3	20	3
3	12	1	1	18	2
4	12	1	3 1 2	11 10 12	2 2 2
		2	2	9	1
5	11	1	1	20	2
6	15	1	3	13	1
		2	4	14	2

Equations in LINGO

```
!here the objective function is;
MIN=@ABS(960-364*X121-216*X311-120*X411-220*X511)+@ABS(960-364*X122-144*X412-
108*X422)+@ABS(960-364*X123-224*X133-132*X413-195*X613)+@ABS(960-140*X114-
330*X624);
!subject to tooling constraints;
2*X121+2*X311+2*X411+2*X511<=5;
2*X122+2*X412+X422<=5;
2*X123+X133+2*X413+X613<=5;
X114+2*X624<=5;
```

Solution IN Lingo

Feasible solution found.

Objective value:	235.0000
Infeasibilities:	0.000000
Total solver iterations:	57
Model Class:	NLP
Total variables:	13
Nonlinear variables:	13
Integer variables:	0
Total constraints:	5
Nonlinear constraints:	1
Total nonzeros:	26
Nonlinear nonzeros:	13

Variable	Value	Reduced Cost
X121	2.500000	0.000000
X311	0.000000	148.0000
X411	0.000000	244.0000
X511	0.000000	144.0000
X122	2.500000	0.000000
X412	0.000000	220.0000
X422	0.000000	74.00000
X123	1.788670	-84.00000
X133	1.086394	0.000000
X413	0.000000	-316.0000
X613	0.3362649	-29.00000
X114	0.000000	25.00000
X624	2.500000	0.000000

Row	Slack or Surplus	Dual Price
1	235.0000	-1.000000
2	0.000000	182.0000
3	0.000000	182.0000
4	0.000000	-224.0000

Solution Obtained BY AIS

THE INITIAL ANTIBODY POPULATION

1	3	2	4	5	6	0.02439	0.05564	1	-41.00000	65
3	6	4	2	1	5	0.02439	0.05564	1	-41.00000	65
2	1	6	3	4	5	0.02439	0.05564	1	-41.00000	65
2	5	3	4	6	1	0.00383	0.00874	0	-261.00000	76
4	2	1	5	3	6	0.00100	0.00227	0	1004.00000	47
3	4	6	1	2	5	0.11111	0.25348	5	-9.00000	64
3	5	4	2	6	1	0.00100	0.00227	0	1004.00000	47
2	6	4	1	3	5	0.05882	0.13419	2	-17.00000	65
4	2	1	3	6	5	0.00100	0.00227	0	1004.00000	47
2	5	1	6	3	4	0.00383	0.00874	0	-261.00000	76
5	4	3	2	1	6	0.00100	0.00227	0	1004.00000	47
1	6	4	5	3	2	0.02222	0.05070	1	-45.00000	64
1	5	6	2	4	3	0.02222	0.05070	1	-45.00000	64
5	6	4	3	1	2	0.00383	0.00874	0	-261.00000	76
5	6	4	1	3	2	0.02222	0.05070	1	-45.00000	64
1	2	3	6	4	5	0.02439	0.05564	1	-41.00000	65
5	2	6	1	3	4	0.00383	0.00874	0	-261.00000	76
6	4	3	1	5	2	0.05882	0.13419	2	-17.00000	65
6	5	4	1	2	3	0.02222	0.05070	1	-45.00000	64
5	1	3	6	2	4	0.00383	0.00874	0	-261.00000	76

THE CLONE POPULATION AFTER ADDING NEW POPULATION

4	6	3	2	1	5	0.11111	0.05564	1	-9.00000	64
3	5	1	6	4	2	0.11111	0.00874	0	-9.00000	64
2	3	6	1	4	5	0.05882	0.05564	1	-17.00000	65
6	4	3	2	5	1	0.05882	0.00874	0	-17.00000	65
2	4	1	5	6	3	0.04762	0.00874	0	-21.00000	64
2	1	3	6	5	4	0.02439	0.13419	2	-41.00000	65
1	6	3	5	2	4	0.02439	0.05070	1	-41.00000	65
1	2	4	5	6	3	0.02222	0.00874	0	-45.00000	64
4	1	2	5	3	6	0.00483	0.13419	2	207.00000	52
4	6	1	2	3	5	0.00474	0.05070	1	211.00000	53
3	2	4	5	6	1	0.00383	0.00874	0	-261.00000	76
5	2	4	3	6	1	0.00383	0.05564	1	-261.00000	76
5	3	4	2	1	6	0.00100	0.05070	1	1004.00000	47
3	1	5	4	6	2	0.11111	0.05564	1	-9.00000	64
6	3	5	4	1	2	0.00383	0.05564	1	-261.00000	76
6	4	2	3	5	1	0.05882	0.05564	1	-17.00000	65
1	5	4	6	2	3	0.02222	0.00874	0	-45.00000	64
2	1	3	6	5	4	0.02439	0.13419	2	-41.00000	65
6	2	3	4	5	1	0.00383	0.05070	1	-261.00000	76
6	5	3	2	1	4	0.00383	0.00874	0	-261.00000	76

THE ITERATION NO. 200

THE CLONE POPULATION

Sequence	affinity	affinity Index	System unbalance	Throwput	
4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
6 3 5 1 4 2	0.11111	0.05564	1	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
6 4 3 2 5 1	0.05882	0.00874	0	-17.00000	65
2 4 1 5 6 3	0.04762	0.00874	0	-21.00000	64
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
1 2 4 5 6 3	0.02222	0.00874	0	-45.00000	64
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
3 2 4 5 6 1	0.00383	0.00874	0	-261.00000	76
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
3 5 6 4 1 2	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
1 6 5 2 4 3	0.02222	0.00874	0	-45.00000	64
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47
3 5 1 6 4 2	0.11111	0.00874	0	-9.00000	64
4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
6 3 5 1 4 2	0.11111	0.05564	1	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
3 5 6 4 1 2	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47

THE CLONE POPULATION AFTER EX- & INV_MUTATION

4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
3 6 5 1 4 2	0.11111	0.05564	1	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
6 4 3 2 5 1	0.05882	0.00874	0	-17.00000	65
2 4 1 5 6 3	0.04762	0.00874	0	-21.00000	64
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
1 2 4 5 6 3	0.02222	0.00874	0	-45.00000	64
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
3 2 4 5 6 1	0.00383	0.00874	0	-261.00000	76
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
3 5 6 4 1 2	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 1 5 2 4 3	0.02222	0.00874	0	-45.00000	64
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47
3 5 1 6 4 2	0.11111	0.00874	0	-9.00000	64
4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
6 3 5 1 4 2	0.11111	0.05564	1	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
1 2 4 5 6 3	0.02222	0.00874	0	-45.00000	64
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
3 2 4 5 6 1	0.00383	0.00874	0	-261.00000	76
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
3 5 6 4 1 2	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 1 5 2 4 3	0.02222	0.00874	0	-45.00000	64
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47
3 5 1 6 4 2	0.11111	0.00874	0	-9.00000	64
4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
6 3 5 1 4 2	0.11111	0.05564	1	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
3 5 6 4 1 2	0.00383	0.05564	1	-261.00000	76
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47

THE CLONE POPULATION AFTER SORTING

4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
3 6 5 1 4 2	0.11111	0.05564	1	-9.00000	64
4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
3 4 6 5 1 2	0.11111	0.05564	1	-9.00000	64
3 5 1 6 4 2	0.11111	0.00874	0	-9.00000	64
6 3 5 1 4 2	0.11111	0.05564	1	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
6 4 3 2 5 1	0.05882	0.00874	0	-17.00000	65
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
2 4 3 5 6 1	0.05882	0.05564	1	-17.00000	65
2 4 1 5 6 3	0.04762	0.00874	0	-21.00000	64
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
1 3 2 6 4 5	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
1 6 4 2 3 5	0.02439	0.05070	1	-41.00000	65
1 2 4 5 6 3	0.02222	0.00874	0	-45.00000	64
6 1 5 2 4 3	0.02222	0.00874	0	-45.00000	64
2 1 4 5 3 6	0.02222	0.13419	2	-45.00000	64
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
3 2 4 5 6 1	0.00383	0.00874	0	-261.00000	76
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
3 5 6 4 1 2	0.00383	0.05564	1	-261.00000	76
6 3 2 5 1 4	0.00383	0.05564	1	-261.00000	76
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47

THE CLONE POPULATION AFTER RECEPT. EDITING

4 6 3 2 1 5	0.11111	0.05564	1	-9.00000	64
3 5 1 6 4 2	0.11111	0.00874	0	-9.00000	64
2 3 6 1 4 5	0.05882	0.05564	1	-17.00000	65
6 4 3 2 5 1	0.05882	0.00874	0	-17.00000	65
2 4 1 5 6 3	0.04762	0.00874	0	-21.00000	64
2 1 3 6 5 4	0.02439	0.13419	2	-41.00000	65
1 6 3 5 2 4	0.02439	0.05070	1	-41.00000	65
1 2 4 5 6 3	0.02222	0.00874	0	-45.00000	64
4 1 2 5 3 6	0.00483	0.13419	2	207.00000	52
4 6 1 2 3 5	0.00474	0.05070	1	211.00000	53
3 2 4 5 6 1	0.00383	0.00874	0	-261.00000	76
5 2 4 3 6 1	0.00383	0.05564	1	-261.00000	76
5 3 4 2 1 6	0.00100	0.05070	1	1004.00000	47

rpop= 13

Sample Problem 3.

Job(i)	Batch size(b)	Operation Number(k)	Machine Number(j)	Unit Processing time(in Mins)	Tool slots available(t_j)
1	10	1 2 3	1 2 2	11 14 10	1 3 2
2	9	1 2 3	4 2 3	9 9 6	1 1 2
3	12	1 2 3	2 3 2	14 12 12	2 1 1
4	16	1	1	15	2
5	6	1 2 2	2 1 4	10 12 7	1 2 1
6	8	1	4	12	

Euation In LINGO

```
!here is the objective function;
MIN=@ABS(960-110*X111-240*X411-72*X521)+@ABS(960-140*X122-100*X132-81*X212-
168*X312-144*X332-60*X512)+@ABS(960-140*X123-81*X213-54*X223-168*X313-
144*X323)+@ABS(960-81*X214-42*X534-96*X614);
!subject to tooling constraint;
X111+2*X411+2*X521<=5;
3*X122+2*X132+X212+2*X312+X332+2*X512<=5;
3*X123+X213+2*X223+2*X313+X323<=5;
X214+X534+X614<=5;
```

Solution in LINGO

```
Local optimal solution found.
  Objective value:                              1320.000
  Infeasibilities:                              0.000000
  Total solver iterations:                            21

  Model Class:                                       NLP

  Total variables:                    17
  Nonlinear variables:                17
  Integer variables:                   0

  Total constraints:                   5
  Nonlinear constraints:               1

  Total nonzeros:                     34
  Nonlinear nonzeros:                 17

                                Variable           Value        Reduced Cost
                                    X111        0.000000            10.00000
                                    X411        2.500000            0.000000
                                    X521        0.000000            168.0000
                                    X122        0.000000            292.0000
                                    X132        0.000000            188.0000
                                    X212        0.000000            63.00000
                                    X312        0.000000            120.0000
                                    X332        5.000000            0.000000
                                    X512        0.000000            228.0000
                                    X123        0.000000            292.0000
                                    X213        0.000000            63.00000
                                    X223        0.000000            234.0000
                                    X313        0.000000            120.0000
                                    X323        5.000000            0.000000
                                    X214        0.000000            15.00000
                                    X534        0.000000            54.00000
                                    X614        5.000000            0.000000

                                     Row    Slack or Surplus      Dual Price
                                       1        1320.000           -1.000000
                                       2        0.000000            120.0000
                                       3        0.000000            144.0000
                                       4        0.000000            144.0000
                                       5        0.000000            96.00000
```

Solution Obtained By AIS

```
THE INITIAL ANTIBODY POPULATION
  1  3  2  4  5  6   0.00129   0.05931    1   778.00000   46
  3  6  4  2  1  5   0.00122   0.05634    1   819.00000   51
  2  1  6  3  4  5   0.00091   0.04199    0  1099.00000   43
  2  5  3  4  6  1   0.00122   0.05634    1   819.00000   51
  4  2  1  5  3  6   0.00091   0.04199    0  1099.00000   43
  3  4  6  1  2  5   0.00129   0.05931    1   778.00000   46
  3  5  4  2  6  1   0.00122   0.05634    1   819.00000   51
  2  6  4  1  3  5   0.00091   0.04199    0  1099.00000   43
  4  2  1  3  6  5   0.00091   0.04199    0  1099.00000   43
  2  5  1  6  3  4   0.00086   0.03961    0  1165.00000   33
  5  4  3  2  1  6   0.00122   0.05634    1   819.00000   51
  1  6  4  5  3  2   0.00129   0.05931    1   778.00000   46
  1  5  6  2  4  3   0.00086   0.03961    0  1165.00000   33
  5  6  4  3  1  2   0.00122   0.05634    1   819.00000   51
  5  6  4  1  3  2   0.00122   0.05634    1   819.00000   51
  1  2  3  6  4  5   0.00091   0.04199    0  1099.00000   43
  5  2  6  1  3  4   0.00086   0.03961    0  1165.00000   33
  6  4  3  1  5  2   0.00129   0.05931    1   778.00000   46
  6  5  4  1  2  3   0.00122   0.05634    1   819.00000   51
  5  1  3  6  2  4   0.00086   0.03961    0  1165.00000   33
```

```
rpop=    7
THE CLONE POPULATION AFTER ADDING NEW POPULATION
  1  3  2  4  5  6   0.00129   0.05931    1   778.00000   46
  3  1  4  6  2  5   0.00129   0.04199    0   778.00000   46
  6  5  2  1  4  3   0.00122   0.05634    1   819.00000   51
  2  3  6  5  1  4   0.00122   0.05634    1   819.00000   51
  1  2  4  5  3  6   0.00091   0.03961    0  1099.00000   43
  6  4  2  1  5  3   0.00091   0.05634    1  1099.00000   43
  1  5  2  6  3  4   0.00086   0.05634    1  1165.00000   33
  2  1  3  5  4  6   0.00086   0.04199    0  1165.00000   33
  5  4  3  1  6  2   0.00122   0.04199    0   819.00000   51
  1  3  6  2  5  4   0.00129   0.03961    0   778.00000   46
  2  1  6  4  3  5   0.00091   0.05634    1  1099.00000   43
  2  3  6  5  1  4   0.00122   0.05931    1   819.00000   51
  6  5  1  2  4  3   0.00086   0.03961    0  1165.00000   33
  5  2  4  1  6  3   0.00122   0.05634    1   819.00000   51
  6  2  4  3  5  1   0.00122   0.05634    1   819.00000   51
  2  4  5  3  1  6   0.00122   0.04199    0   819.00000   51
  4  5  1  2  6  3   0.00122   0.03961    0   819.00000   51
  1  4  5  6  2  3   0.00091   0.05931    1  1099.00000   43
  3  5  6  1  2  4   0.00122   0.05634    1   819.00000   51
  1  6  5  2  3  4   0.00086   0.03961    0  1165.00000   33
```

THE ITERATION NO. 63

```
THE CLONE POPULATION
  1  3  2  4  5  6   0.00129   0.05931    1   778.00000   46
  3  1  4  6  2  5   0.00129   0.04199    0   778.00000   46
  6  5  2  1  4  3   0.00122   0.05634    1   819.00000   51
  6  4  2  1  5  3   0.00091   0.05931    1  1099.00000   43
  6  4  2  1  5  3   0.00091   0.05634    1  1099.00000   43
  1  5  2  6  3  4   0.00086   0.05634    1  1165.00000   33
  4  1  5  3  6  2   0.00129   0.05634    1   778.00000   46
  5  3  6  2  1  4   0.00122   0.04199    0   819.00000   51
  5  1  4  6  2  3   0.00086   0.04199    0  1165.00000   33
  2  5  1  4  3  6   0.00086   0.03961    0  1165.00000   33
  6  1  4  2  3  5   0.00091   0.05634    1  1099.00000   43
  3  5  1  4  6  2   0.00122   0.05931    1   819.00000   51
  4  2  5  6  3  1   0.00122   0.03961    0   819.00000   51
  2  3  6  5  1  4   0.00122   0.05634    1   819.00000   51
  6  2  5  1  4  3   0.00086   0.05634    1  1165.00000   33
  1  3  2  5  4  6   0.00129   0.04199    0   778.00000   46
  5  6  4  1  2  3   0.00122   0.03961    0   819.00000   51
  1  5  4  3  6  2   0.00086   0.05931    1  1165.00000   33
  4  6  3  1  2  5   0.00129   0.05634    1   778.00000   46
  1  2  4  5  3  6   0.00091   0.03961    0  1099.00000   43
  1  3  2  4  5  6   0.00129   0.05931    1   778.00000   46
  6  5  2  1  4  3   0.00122   0.05634    1   819.00000   51
  6  4  2  1  5  3   0.00091   0.05931    1  1099.00000   43
  6  4  2  1  5  3   0.00091   0.05634    1  1099.00000   43
  1  5  2  6  3  4   0.00086   0.05634    1  1165.00000   33
  4  1  5  3  6  2   0.00129   0.05634    1   778.00000   46
  6  1  4  2  3  5   0.00091   0.05634    1  1099.00000   43
  3  5  1  4  6  2   0.00122   0.05931    1   819.00000   51
  2  3  6  5  1  4   0.00122   0.05634    1   819.00000   51
  6  2  5  1  4  3   0.00086   0.05634    1  1165.00000   33
  1  5  4  3  6  2   0.00086   0.05931    1  1165.00000   33
  4  6  3  1  2  5   0.00129   0.05634    1   778.00000   46
THE CLONE POPULATION AFTER EX- & INV_MUTATION
  1  3  2  4  5  6   0.00129   0.05931    1   778.00000   46
  3  1  4  6  2  5   0.00129   0.04199    0   778.00000   46
  6  5  2  1  4  3   0.00122   0.05634    1   819.00000   51
  6  4  5  1  2  3   0.00122   0.05931    1   819.00000   51
  6  4  2  1  5  3   0.00091   0.05634    1  1099.00000   43
```

```
  1   5   2   6   3   4   0.00086   0.05634    1  1165.00000   33
  4   1   5   3   6   2   0.00129   0.05634    1   778.00000   46
  6   3   5   2   1   4   0.00122   0.04199    0   819.00000   51
  5   1   4   6   2   3   0.00086   0.04199    0  1165.00000   33
  5   2   1   4   3   6   0.00086   0.03961    0  1165.00000   33
  6   3   2   4   1   5   0.00122   0.05634    1   819.00000   51
  3   1   5   4   6   2   0.00129   0.05931    1   778.00000   46
  5   2   4   6   3   1   0.00122   0.03961    0   819.00000   51
  2   3   6   5   1   4   0.00122   0.05634    1   819.00000   51
  6   5   2   1   4   3   0.00122   0.05634    1   819.00000   51
  1   3   2   5   4   6   0.00129   0.04199    0   778.00000   46
  5   6   4   1   2   3   0.00122   0.03961    0   819.00000   51
  1   5   6   3   4   2   0.00086   0.05931    1  1165.00000   33
  4   6   3   1   2   5   0.00129   0.05634    1   778.00000   46
  1   2   4   5   3   6   0.00091   0.03961    0  1099.00000   43
  1   3   2   4   5   6   0.00129   0.05931    1   778.00000   46
  6   5   2   1   4   3   0.00122   0.05634    1   819.00000   51
  6   4   2   1   5   3   0.00091   0.05931    1  1099.00000   43
  6   4   2   5   1   3   0.00122   0.05634    1   819.00000   51
  1   5   2   6   3   4   0.00086   0.05634    1  1165.00000   33
  4   1   5   3   6   2   0.00129   0.05634    1   778.00000   46
  6   1   4   2   3   5   0.00091   0.05634    1  1099.00000   43
  5   3   1   4   6   2   0.00122   0.05931    1   819.00000   51
  6   3   2   5   1   4   0.00122   0.05634    1   819.00000   51
  6   2   5   1   4   3   0.00086   0.05634    1  1165.00000   33
  1   4   5   3   6   2   0.00129   0.05931    1   778.00000   46
  4   6   3   1   2   5   0.00129   0.05634    1   778.00000   46


THE CLONE POPULATION AFTER SORTING
  1   3   2   4   5   6   0.00129   0.05931    1   778.00000   46
  4   1   5   3   6   2   0.00129   0.05634    1   778.00000   46
  1   3   2   5   4   6   0.00129   0.04199    0   778.00000   46
  4   6   3   1   2   5   0.00129   0.05634    1   778.00000   46
  1   3   2   4   5   6   0.00129   0.05931    1   778.00000   46
  4   1   5   3   6   2   0.00129   0.05634    1   778.00000   46
  4   6   3   1   2   5   0.00129   0.05634    1   778.00000   46
  3   1   4   6   2   5   0.00129   0.04199    0   778.00000   46
  3   1   5   4   6   2   0.00129   0.05931    1   778.00000   46
  1   4   5   3   6   2   0.00129   0.05931    1   778.00000   46
  6   5   2   1   4   3   0.00122   0.05634    1   819.00000   51
  6   4   5   1   2   3   0.00122   0.05931    1   819.00000   51
  6   3   5   2   1   4   0.00122   0.04199    0   819.00000   51
  6   3   2   4   1   5   0.00122   0.05634    1   819.00000   51
  5   2   4   6   3   1   0.00122   0.03961    0   819.00000   51
  6   5   2   1   4   3   0.00122   0.05634    1   819.00000   51
  5   6   4   1   2   3   0.00122   0.03961    0   819.00000   51
  6   5   2   1   4   3   0.00122   0.05634    1   819.00000   51
  6   4   2   5   1   3   0.00122   0.05634    1   819.00000   51
  5   3   1   4   6   2   0.00122   0.05931    1   819.00000   51
  6   3   2   5   1   4   0.00122   0.05634    1   819.00000   51
  2   3   6   5   1   4   0.00122   0.05634    1   819.00000   51
  1   2   4   5   3   6   0.00091   0.03961    0  1099.00000   43
  6   4   2   1   5   3   0.00091   0.05931    1  1099.00000   43
  6   1   4   2   3   5   0.00091   0.05634    1  1099.00000   43
  6   4   2   1   5   3   0.00091   0.05634    1  1099.00000   43
  1   5   2   6   3   4   0.00086   0.05634    1  1165.00000   33
  5   1   4   6   2   3   0.00086   0.04199    0  1165.00000   33
  5   2   1   4   3   6   0.00086   0.03961    0  1165.00000   33
  1   5   6   3   4   2   0.00086   0.05931    1  1165.00000   33
  1   5   2   6   3   4   0.00086   0.05634    1  1165.00000   33
  6   2   5   1   4   3   0.00086   0.05634    1  1165.00000   33
THE CLONE POPULATION AFTER RECEPT. EDITING
  1   3   2   4   5   6   0.00129   0.05931    1   778.00000   46
  3   1   4   6   2   5   0.00129   0.04199    0   778.00000   46
  6   5   2   1   4   3   0.00122   0.05634    1   819.00000   51
  2   3   6   5   1   4   0.00122   0.05634    1   819.00000   51
  1   2   4   5   3   6   0.00091   0.03961    0  1099.00000   43
  6   4   2   1   5   3   0.00091   0.05634    1  1099.00000   43
  1   5   2   6   3   4   0.00086   0.05634    1  1165.00000   33
```

References

1) *N. Nagarjuna, O. Mahesh, K. Rajagopal; "A heuristic based on multi-stage programming approach for machine-loading problem in a flexible manufacturing system"*;*Robotics and Computer-Integrated Manufacturing 22 (2006) 342–352.ScienceDirect*

2) *Akhilesh Kumar , Prakash , M.K. Tiwari , Ravi Shankar , Alok Baveja; "Solving machine-loading problem of a flexible manufacturing system with constraint-based genetic algorithm"; European Journal of Operational Research 175 (2006) 1043–1069*

3) *S. K. Mukhopadhyay; M. K. Singh; R. Srivastava; "FMS machine loading: A simulated annealing approach"; International Journal of Production Research.*

4) *S. G. Ponnambalam,Low Seng Kiat; "Solving Machine Loading Problem in Flexible Manuf. Systems Using PSO"; World Academy of Science,Engineering and Technology.39 2008*

5) *Felix T.* **S.** *Chan, Rahul Swamkar and Manoj K. Tiwari; "A Random Search Approach to the Machine Loading Problem of an FMS"; Intemational Symposium on Intelligent Control Taipei, Taiwan, September 2-4,2004.*

6) *N. K. Vidyarthi; M. K. Tiwari; "Machine loading problem of FMS: A fuzzy-based heuristic approach"; International Journal of Production Research.14 Nov, 2010.*

7) *Sandhyarani Biswas, S.S.Mahapatra; "Machine Loading in Flexible Manufacturing System: A Swarm Optimization Approach"; Eighth Int. Conference on Opers. & Quant. Management October 17-20, 2007.*

8) *M. Chandrasekaran . P. Asokan .S. Kumanan . T. Balamurugan . S. Nickolas; "Solving job shop scheduling problems using artificial immune system"; Int J Adv Manuf Technol (2006) 31: 580–593,3 Jan,2006.*

9) Nitesh Khilwania, Anoop Prakashb, Ravi Shankarc, M.K. Tiwari; "Fast clonal algorithm"; Engineering Applications of Artificial Intelligence 21 (2008) 106–128.

10) *Anoop Prakash a, Nitesh Khilwani b, M.K. Tiwari c,*, Yuval Cohen; "Modified immune algorithm for job selection and operation allocation problem in flexible manufacturing systems"; Advances in Engineering Software 39 (2008) 219–232.*

11) *Roby Thomas, Marvin D. Troutt; "Comparison of objective criteria for set-up planning in complementary flexible manufacturing systems" ;Computers & Industrial Engineering 53 (2007) 17–29.*

12) *Santosh Kumar Mandal ; Mayank Kumar Pandey ; M. K. Tiwari ; ":Incorporating dynamism in traditional machine loading problem: an AI-based optimisation approach" International Journal of Production Research,28 May 2009.*

13) *Tiwari MK, Hazarika B, Vidyarthi NK, Jaggi P, Mukhopadyay SK. A heuristic solution approach to the machine-loading problem of an FMS and its Petri net model. Int J Production Res 1997;35.*

14) *Shanker K, Tzen YJ. A loading and dispatching problem in a random flexible manufacturing system. Int J Production Res 1985;23*